Listing of Claims:

1. (Previously Presented) A method for automatically preventing errors in computer software having a plurality of different life cycle phases, the method comprising:

storing source code of the computer software in a code repository;

executing a plurality of software verification tools to verify the computer software, wherein each of the plurality of software verification tools corresponds to a respective lifecycle phase of the computer software, and automatically generates one or more test cases from the source code of the computer software;

generating verification results for each respective lifecycle phase of the computer software, responsive to executing the plurality of software verification tools and the automatically generated test cases; and

processing the verification results for generating a representation of functional behavior of the computer software.

2. (Original) The method of claim 1 further comprising providing a common configuration file for the plurality of verification tools.

3. (Previously Presented) The method of claim 2, further comprising customizing a verification scope of one or more of the verification tools by modifying the common configuration file responsive to an objective criterion of quality of the computer software.

4. (Original) The method of claim 2 further comprising modifying a portion of the common configuration file specific to one of the plurality of verification tools responsive to the objective criterion of quality of the computer software.

5. (Original) The method of claim 2 further comprising modifying a portion of the common configuration file specific to one of a plurality of software developers responsive to the objective criterion of quality of the computer software.

6. (Previously Presented) The method of claim 1, further comprising formulating the verification results in a confidence factor represented by the equation:

$$C = p/t \times 100,$$

where p is number of successful test cases and t is total number of test cases.

7. (Original) The method of claim 1, wherein each portion of the computer software being developed by a software developer of a plurality of software developers, and the verification results include a plurality of objective criteria each of the plurality of objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer of the plurality of software developers.

8. (Original) The method of claim 7 further comprising providing a common configuration file for the plurality of verification tools; and modifying the common configuration file responsive to one or more objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer.

9. (Original) The method of claim 7 further comprising verifying a first portion of the computer software developed by a first developer of the plurality of software developers using the plurality of verification tools, before the computer software is stored in the code repository.

10. (Original) The method of claim 9 further comprising allowing storing the first portion of the computer software in the code repository only if result of verification of the first portion meets a set standard.

11. (Original) The method of claim 10 further comprising modifying the set standard responsive to the objective criterion of quality of the computer software.

12. (Original) The method of claim 10, wherein the set standard is common to each of the plurality of software developers.

13. (Original) The method of claim 10, wherein the set standard is unique to at least one of the plurality of software developers.

14. (Previously Presented) A system for automatically preventing errors in computer software having a plurality of different life cycle phases comprising:

means for storing source code of the computer software in a code repository;

means for executing a plurality of software verification tools to verify the computer software, wherein each of the plurality of software verification tools corresponds to a respective lifecycle phase of the computer software, and generates one or more test cases from the source code of the computer software;

means for generating verification results for each respective lifecycle phase of the computer software, responsive to executing the plurality of software verification tools and the automatically generated test cases; and

means for processing the verification results for generating a representation of functional behavior of the computer software.

15. (Original) The system of claim 14 further comprising means for providing a common configuration file for the plurality of verification tools.

16. (Previously Presented) The system of claim 15 further comprising means for modifying the common configuration file responsive to an objective criterion of quality of the computer software.

17. (Previously Presented) The system of claim 15 further comprising means for modifying a portion of the common configuration file specific to one of the plurality of verification tools responsive to an objective criterion of quality of the computer software.

18. (Previously resente)d e system of claim 15 further comprising means for modifying a portion of the common configuration file specific to one of a plurality of software developers responsive to an objective criterion of quality of the computer software.

19. (Previously Presented) The system of claim 14, further comprising means for formulating the verification results in a confidence factor represented by the equation:

$$C = p/t \times 100,$$

where p is number of successful test cases and t is total number of test cases.

20. (Original) The system of claim 14, wherein each portion of the computer software being developed by a software developer of a plurality of software developers, and the verification results include a plurality of objective criteria each of the plurality of objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer of the plurality of software developers.

21. (Original) The system of claim 20 further comprising means for providing a common configuration file for the plurality of verification tools; and means for modifying the common configuration file responsive to one or more objective criteria corresponding to quality of a respective portion of the computer software developed by each software developer.

22. (Original) The system of claim 20 further comprising means for verifying a first portion of the computer software developed by a first developer of the plurality of software

developers using the plurality of verification tools, before the computer software is stored in the code repository.

23. (Original) The system of claim 22 further comprising means for allowing storing the first portion of the computer software in the code repository only if result of verification of the first portion meets a set standard.

24. (Original) The system of claim 23 further comprising means for modifying the set standard responsive to the objective criterion of quality of the computer software.

25. (Original) The system of claim 23, wherein the set standard is common to each of the plurality of software developers.

26. (Original) The system of claim 23, wherein the set standard is unique to at least one of the plurality of software developers.

27. (Previously Presented) A method for automatically preventing errors in computer software having a plurality of different life cycle phases, the method comprising:

providing a known error in the computer software, the known error belonging to a class of errors;

providing a plurality of software verification tools each of the plurality of software verification tools corresponding to a respective lifecycle phase of the computer software;

analyzing the known error in the computer software to determine what phase of the lifecycle the error was introduce;

customizing a verification scope of one or more of the plurality of verification tools that correspond to the lifecycle phase that the known error was introduced; and

executing the plurality of software verification tools to verify the known error is detected in computer software.

28. (Previously Presented) The method of claim 27, further comprising customizing the verification scope by modifying a configuration file common to the verification tools based on an objective criterion of quality of the computer software.

29. (Original) The method of claim 28 further comprising modifying a portion of the configuration file specific to one of the plurality of verification tools based on the objective criterion of quality of the computer software.

30. (Original) The method of claim 28 further comprising modifying a portion of the common configuration file specific to one of a plurality of software developers responsive to the objective criterion of quality of the computer software.

31. (Previously Presented) The method of claim 27, further comprising processing the verification results for generating an objective criterion of quality of the computer software by formulating the verification results in a confidence factor represented by the equation:

$$C = p/t \times 100,$$

where p is number of successful test cases and t is total number of test cases.

32. - 42 (Canceled)

43. (Previously Presented) The method of claim 28 further comprising customizing the verification scope of one or more of the plurality of verification tools for a second time, if the known error is not detected by executing the plurality of software verification tools.

44. (Previously Presented) The method of claim 27 further comprising executing the plurality of software verification tools periodically to prevent the known error from re-occurring when the computer software is modified.

45. (Previously Presented) A system for automatically preventing errors in computer software having a plurality of different life cycle phases comprising:

means for providing a known error in the computer software, the known error belonging to a class of errors;

means for providing a plurality of software verification tools each of the plurality of software verification tools corresponding to a respective lifecycle phase of the computer software;

means for analyzing the known error in the computer software to determine what phase of the lifecycle the error was introduce; and

means for customizing a verification scope of one or more of the plurality of verification tools that correspond to the lifecycle phase that the known error was introduced.

46. (Previously Presented) The system of claim 45 further comprising means for executing the plurality of software verification tools to verify the known error is detected in computer software.

47. (Previously Presented) The system of claim 46 further comprising means for customizing the verification scope of one or more of the plurality of verification tools for a second time, if the known error is not detected by executing the plurality of software verification tools.

48. (Previously Presented) The system of claim 45 further comprising means for executing the plurality of software verification tools periodically to prevent the known error from re-occurring when the computer software is modified.